Bachelorarbeitpräsentation Segmentierung von Punktwolken mit Neuronalen Netzen, Kurzfassung

Willkommen zu meiner Bachelorarbeitpräsentation zum Thema Segmentierung von Punktwolken mit Neuronalen Netzen.

(Folie 2)

Motivation der Arbeit war vor allem das Thema autonomes Fahren, denn wie in diesem Paper bemerkt wird, ist das Erkennen der Umwelt und der Möglichkeiten zur Fortbewegung dafür eine Hauptzutat. Aber auch für zahlreiche andere moderne Entwicklungsthemen ist das maschinelle Sehen, zu dem auch die Segmentierung gehört, ein wichtiger Teil der Prozesskette, wie beispielsweise bei Augmented Reality und autonomer Robotik.

(Folie 3)

In den letzten Jahren wurden mit Deep Learning enorme Fortschritte im Bereich der Bildsegmentierung gemacht. Die Erkennung der Umgebung ist aber letztendlich ein dreidimensionales Problem. Ziel der Arbeit ist es deshalb, nicht nur Bilder, sondern auch Punktwolken als räumliche Strukturen zu verarbeiten.

Die Segmentierung von Bildern und Punktwolken ist immer noch eine offene Aufgabe. Auch wenn einige Arbeiten dieses Thema erfolgreich bearbeiten konnten, ist es weiterhin problematisch, die Laufzeit der Prozesse zu reduzieren und dabei hoch qualitative Ergebnisse zu erzeugen. Gerade für autonomes Fahren ist Echtzeitfähigkeit eine notwendige Eigenschaft.

Speziell das Verarbeiten von Punktwolken mit Neuronalen Netzen ist ein Problem, da es sich dabei um ungeordnete Strukturen handelt.

(Folie 4)

Aufgrund dieser Problemstellung verfolgt die Arbeit folgende Ziele. Zuerst sollte ein System zur semantischen Segmentierung von Bildern und Punktwolken entwickelt werden. Dann sollten geeignete Datensätze für die Durchführung von Experimenten ausgewählt werden und Experimente durchgeführt werden. Die Ziele waren dabei die Optimierung der Laufzeit und Verbesserung der Ergebnisse.

(Folie 5)

Die Arbeit befasst sich mit semantischer Segmentierung. Semantische Segmentierung ist ein Prozess, bei dem jedem Pixel in einem Bild eine valide Klasse zugeteilt wird. Es wird dabei nicht zwischen Instanzen von zählbaren Objekten unterschieden, wie auf dem Bild zu sehen ist. Die Anzahl der Labels ist deshalb bei jedem Bild gleich, was für den Einsatz von Neuronalen Netzen vorteilhaft ist. In dieser Arbeit wird der Begriff auf Punktwolken übertragen und so verstanden, dass jeder Punkt in zumindest einem bestimmten Bereich der Punktwolke ein Label erhält.

(Folie 6)

Eine aktuelle Arbeit zur Bildsegmentierung mit Neuronalen Netzen ist beispielsweise UPSNet, das eine panoptische Segmentierung von Bildern durchführt. Im Gegensatz zur semantischen Segmentierung wird dabei zwischen Instanzen von zählbaren Objekten unterschieden.

Das derzeit am besten gewertete Modell zur semantischen Segmentierung ist DeepLab, das auf Convolutional Neural Networks basiert. Es ist die Architektur, die in dieser Arbeit verwendet wird und ich werde später noch darüber reden.

(Folie 7)

Ein Ansatz zur Verarbeitung von Punktwolken mit Neuronalen Netzen ist PointNet. Dabei handelt es sich um eine Architektur, die eine Punktwolke als Eingabe erhält und entweder eine Klasse für den gesamten Input oder ein Label für jeden Punkt ausgibt. Das Problem der ungeordneten Daten wird darin so gelöst, dass zuerst eine globale Signatur aus den Eingabedaten erstellt wird. Für die Aufgabe der Segmentierung wird die gloabel Signatur mit den lokalen Informationen kombiniert.

(Folie 8)

Ich werde als nächstes auf die Grundlagentheorie der Arbeit eingehen.

Ein Convolutional Neural Network, kurz CNN, ist ein Neuronales Netz, das in mindestens einer Schicht eine Faltungsoperation durchführt. Meistens enthalten solche Netze auch s.g. Pooling Layers, die die Daten zusammenfassen und somit die Datenmenge für die späteren Schichten verringern und Invarianz gegenüber kleinen Änderungen erreichen. Die letzte Schicht eines CNN ist häufig ein Fully Connected Layer, wie sie in MLPs vorkommen. Ein Netz, das auf Fully Connected Layers verzichtet und nur Faltungsoperationen durchführt wird als Fully Convolutional Network bezeichnet.

(Folie 9)

CNNs sind für die Bildverarbeitung aus folgenden Gründen besonders geeignet. Die Größe des Faltungskernels kann frei gewählt werden und ist in der Regel vernachlässigbar klein verglichen mit den verarbeiteten Datenmenge, was CNNs besonders Speichereffizient macht. Außerdem wird dadurch Equivarianz bezüglich Translation erreicht. Das bedeutet, wenn die Eingabedaten verschoben sind, tritt die gleiche Verschiebung den Ausgabedaten auf. Außerdem kann eine Fully Convolutional CNN Bilder in unterschiedlichen Größen verarbeiten.

(Folie 10)

In CNNs wird normalerweise keine Faltung im streng mathematischen Sinn, sonder eine Abwandlung eingesetzt. Eine dieser Variationen ist Atrous Convolution. Sie entspricht der Faltung mit eine spärlich bestückten Kernel bzw. einem Kernel mit Löchern, wie in dem Bild dargestellt. Die Abstände zwischen zu berücksichtigenden Werten wird durch die s.g. Erweiterungsrate angegeben. Durch Atrous Convolution erhält man ein größeres Sichtfeld der Kernel bei gleichem Rechenaufwand und eine höhere Auflösung als beim Einsatz von Pooling und Interpolation.

(Folie 11)

Um die Ergebnisse von Bildsegmentierung vor allem im Bereich von Kanten zu verbessern, können Conditional Random Fields, kurz CRFs eingesetzt werden. Dabei handelt es sich um Neuronale Netze, die nicht eine Funktion lernen, die eine Eingabe x auf eine Ausgabe y abbildet, sondern stattdessen eine Wahrscheinlichkeitsverteilung annähert, die beschreibt, mit welcher Wahrscheinlichkeit eine Ausgabe aus einer Eingabe in einem Prozess entstanden ist. Die Wahrscheinlichkeitsverteilung wird normalerweise als Markov-Kette erster Ordnung modeliert. Unter dieser Annahme, die ist Funktion mit maximalem Informationsgehalt die Exponentialfunktion und eine Ausgabe kann aus einer Eingabe mit dem Viterbi-Algorithmus berechnet werden. Das Lern-Verfahren beruht auf Gradient Descend, wobei die Berechnung des Gradienten aufwändig ist und er oft mit dem Newton-Verfahren angenähert wird.

(Folie 12)

In dieser Arbeit spielt nicht nur Neuronale Netze, sondern auch Kamerprojektion eine wichtige Rolle. Eine Kamera ist ein Werkzeug, das es erlaubt, dreidimensionale Objekte auf einer zweidimensionalen Fläche abzubilden. Die Funktionsweise kann am Modell der Lochkamera anschaulich erklärt werden. Darin verläuft, wie in der Abbildung dargestellt, ein Strahl von einem Punkt M durch ein Projektionszentrum C und schneidet eine Bildebene in einem Bildpunkt m. Dann ist der Bildpunkt m die Projektion des realen Punktes M. Diese Projektion führt zu einem Verlust an Informationen und zwar über Entfernungen, Längen, Winkel, Verhältnisse und dementsprechend Formen. Um von einem Bild auf die Realität schließen zu können ist deswegen Kalibrierung notwendig.

(Folie 13)

Kamerakalibrierung ermöglicht, zu berechen, wo ein Punkt im dreidimensionalen Raum auf der Bildebene abgebildet wird.

Außerdem lässt sich dadurch aus zwei Punkten im Bild der Winkel zwischen den Strahlen bestimmen, aus denen sie projeziert wurden.

Damit kann beispielsweise ermittelt werden, wo ein Punkt in einem Bild auf der Bildebene einer anderen Kamera erscheint.

(Folie 14)

Für die Arbeit wurde ein einfacher Algorithmus verwendet, der hier dargestellt ist. Ein Bild wird mit Hilfe von DeepLab segmentiert und die gefundenen Labels auf eine Punktwolke projiziert.

(Folie 15)

DeepLab ist eine Architektur zur Bildsegmentierung, die auf DCNNs besiert. Es wurde von Google entwickelt und erstmals im Jahr 2015 vorgestellt. Aufgrund der guten Ergebnisse auf den Datensätzen von Cityscapes und Pascal VOC gilt es als State-of-the-art.

(Folie 16)

Der Arbeitsfluss von DeepLab besteht darin, dass ein Eingabebild von einem Backbone zu einer Score Map verarbeitet wird. Diese wird durch Bi-lineare Interpolation auf die Größe des Eingabebildes vergrößert und von einem Fully Connected CRF weiterverarbeitet.

(Folie 17)

Um CNNs für semantische Segmentierung einzusetzen, werden in DeepLab zunächst alle Fully Connected Layers durch Convolutional Layers ersetzt, um ein Fully Convolutional CNN zu bilden. Noch dazu wird in den tieferen Schichten Atrous Convolution an Stelle von Max-Pooling eingesetzt, um die Auflösung der Score Map zu erhöhen. Um Größeninvarianz zu erreichen, wird außerdem Atrous Spatial Pyramid Pooling eingesetzt. Dabei werden mehrere Schichte, die Atrous Convolution mit unterschiedlicher Erweiterungsrate verwenden, im Netz parallel geschaltet. Aus den Ergebnissen dieser Faltungsoperationen wird dann die Ausgabe der Schicht gebildet.

(Folie 18)

Zur Projektion der Ergebnisse der Segmentierung auf eine Punktwolke muss zuerst die Projektionsmatrix berechnet werden. Dazu muss die verwendete Kamera kalibriert sein. Danach iteriert der Algorithmus durch alle Punkte P der Punktwolke und berechnet durch Multiplikation mit der Projektionsmatrix die Koordinaten des projizierten Punktes auf der Bildebene. Danach wird überprüft, ob die Projektion von Punkt P sich auf der Bildfläche der Kamera befindet und ob er sich vor der Kamera befindet. Ist das der Fall, wird dem Punkt das Label des entsprechenden Pixels auf dem Bild zugeteilt.

(Folie 19)

Kommen wir nun zu den Experimenten. Die Ziele der Experimente waren einerseits, zu ermitteln, was von dem Algorithmus unter Einsatz der aktuellen Technik erwartet werden kann. Andererseits sollte versucht werden, die Ergebnisse durch Änderungen beim Lernvorgang zu verbessern. Außerdem sollten die beiden Backbones verglichen werden. Die Implementierung unterstützt zwei Backbones: das leistungsfähigere Xception65 und das ressourcenschonende MobileNetV2, das für mobile Geräte entwickelt wurde. Da in dieser Arbeit Laufzeit Priorität hat, konzentrieren sich die Experimente auf MobileNetV2.

(Folie 20)

Für die Experimente wurden zwei Datensätze ausgewählt: Cityscapes und KITTI.

Cityscapes ist ein Datensatz für autonomes Fahren, der eine relativ große Menge an annotierten Bilden für semantische Segmentierung bietet und deshalb als hauptsächliche Datenquelle für das Training des Netzes dient.

KITTI ist ein Datensatz für mobile Robotik und autonomes Fahren. Darin werden insgesamt 6 Stunden an Aufnahmen geboten und neben der Bilder auch Punktwolken bereitgestellt. Außerdem bietet KITTI 200 für semantische Segmentierung annotierte Bilder.

(Folie 21)

Das erste Experimente diente dazu, die ideale Trainingsdauer für ein Netz zu finden, das MobileNetV2 als Backbone verwendet. Dazu wurden mehrere Model auf dem Trainingsset des Citydatensatzes trainiert und auf dem Evaluierungsdatensatz in Intersection-over-Union-Metrik bewertet. Wie in der Grafik zu sehen ist, liegt die ideale Trainingsdauer bei etwa 25 Epochen. Danach verschlechtern sich die Ergebnisse wieder. Vermutlich liegt das an Überanpassung bzw Overfitting des Netzes.

(Folie 22)

Diese Grafik zeigt die Ergebnisse des Netzes für jede im Datensatz gelabelte Klasse in IoU-Metrik. Wie man sieht, schneidet das Netz am besten bei der Erkennung von amorphen Objekten ab und erzielt außerdem gut Ergebnisse bei großen Objekten, die im Datensatz häufig vorkommen. Die Ergebnisse bei kleinen Objekten sind deutlich schlechter. Es fällt auf, dass das Netz einige Klasse nicht erkennt. Das ist der Fall bei solchen Labels, die im Datensatz selten vorkommen und große Ähnlichkeit mit einer anderen Klasse haben.

(Folie 23)

Dieses Verhalten ist auch in diesen Beispielergebnissen sichtbar. Autos, Straßen und Gebäude werden darin gut erkannt, Fahrräder und Personen weiniger. Die Straßenbahn im untersten Bild wird überhaupt nicht erkannt.

(Folie 24)

Im nächsten Experiment soll das Netz mit MobileNetV2 mit einem verglichen werden, das Xception65 verwendet. Wie man sieht, erzielt Xception durchwegs bessere Ergebnisse und erkennt alle Klassen zumindest teilweise. Allerdings braucht es dafür fast doppelt so lange auf demselben Rechner.

(Folie 25)

Hier sind Beispielergebnisse von dem Netz mit Xception dargestellt. Wie man sieht, sind die Ergebnisse vor allem bei zählbaren Objekten deutlich besser.

(Folie 26)

Das nächste Experiment soll die Auswirkungen von Nachtraining mit Daten aus einem anderen Datensatz untersuchen. Für diesen Zweck werden die 200 annotierten Bilder aus dem KITTI-Datensatz verwendet. Es wurden mehrere Models trainiert und auf je einem Bild aus dem KITTI- und Cityscapes-Datensatz getestet. Wie man in den oberen Bildern sieht, unterscheiden sich die Bilder aus dem KITTI-Datensatz in ihrer Perspektive und Qualität von denen aus Cityscapes.

Unter den Originalbildern sieht man die Ergebnisse von dem Model mit MobileNetV2 aus den vorherigen Versuchen, das für 25 Epochen auf dem Cityscapes-Datensatz trainiert wurde. Während die Qualität der Ergebnisse auf dem Cityscapes-Bild erwartungsgemäß ausfallen, ist sie bei den Ergebnisse aus den KITTI-Daten deutlich niedriger.

Darunter zu sehen sind die Ergebnisse von einem Model, das für eine Epoche mit Daten aus dem KITTI-Datensatz nachtrainiert ist. Die Ergebnisse auf dem KITTI-Bild verbessern sich deutlich, während sich die auf dem Cityscapes-Bild verschlechtern.

(Folie 27)

Weiteres Nachtraining mit den KITTI-Daten verbessern die Ergebnisse auf dem Bild aus KITTI weiter, wenn auch in kleinerem Maße. Zugleich verschlechtern sich dadurch die Ergebnisse auf Cityscapes weiter.

Als letztes wurde ein Model, das 25 Epochen auf Cityscapes und 5 Epochen auf KITTI trainiert wurde noch eine Epoche mit Cityscapes nachtrainiert. Das Ergebnis ist eine deutliche Verbesserung des Ergebnisses auf Cityscapes und eine Verschlechterung auf KITTI. Das Ergebnis auf dem KITTI-Bild ist dabei trotzdem noch deutlich besser als die des ersten Models.

Dieses Experiment zeigt, dass in dem Netz schnell starke Überanpassung auftritt.

(Folie 28)

Für einen weiteren Versuch, wurde der KITTI-Datensatz in einen Trainings- und einen Evaluierungssatz zu je 100 Bildern aufgeteilt. Dadurch wird es möglich, die Models in IoU-Metrik zu bewerten. In diesem Experiment wurden vier Models evaluiert. Model A ist das Model mit MobileNetV2 aus den vorherigen Experimenten, das 25 Epochen auf dem Cityscapes-Datensatz trainiert ist. Model B wurde 24 Epochen mit einer Mischung aus Cityscapes- und KITTI-Daten trainiert. Wie im Diagramm zu sehen, zeigt es deutlich bessere Ergebnisse als Model A auf den KITTI-Daten, während die Bewertung für die Cityscapes-Daten etwa gleich ist. Für Model C wurde Model B eine Epoche lang mit KITTI-Daten nachtrainiert. Wie im vorherigen Experiment tritt auch hier eine deutliche Verbesserung bei KITTI und eine Verschlechterung bei Cityscapes auf. Model D ist Model C, das noch eine weitere Epoche mit den gemischten Daten trainiert wurde. Es schneidet bei beiden Datensätzen besser ab als Model A und B, zeigt aber auf den KITTI-Daten schlechtere Ergebnisse als Model C.

Das Experiment zeigt, dass eine größere Varietät in den Trainingsdaten positive Auswirkungen hat.

(Folie 29)

Hier sieht man die Ergebnisse der gerade vorgestellten Models für die Bilder aus dem letzten Versuch. Deutlich ist vor allem der Unterschied zwischen den Ausgaben von Model A und D und Model A und C.

(Folie 30)

Als letztes soll die Segmentierung von Punktwolken untersucht werden. Für diesen Versuch wird das Model C aus dem letzten Versuch verwendet, das die beste Bewertung auf dem KITTI-Datensatz aufweist. Diese Folie zeigt ein Bild aus dem KITTI-Datensatz und die Ausgabe des Netzes dafür. Wie man sieht, werden Straße und Autos gut erkannt, die LKWs erwartungsgemäß nicht.

(Folie 31)

Hier sieht man die Punktwolke mit den darauf projizierten Labels aus verschiedenen Perspektiven. Offensichtlich hängt die Qualität der segmentierten Punktwolke von der des segmentierten Bildes ab und es können nur Aussagen über Punkte im Sichtfeld der Kamera gemacht werden. Wegen verschwommen erkannter Ränder treten bei weit entfernten Punkten mehr Fehler auf als bei nahen. Die Autos werden, wie auf dem Bild, gut erkannt, während die falschen Labels der LKWs natürlich auch in der Punktwolke wiederzufinden sind.

(Folie 32)

Hier ist ein weiteres Beispiel. Man sieht ein Auto, das vom Netz beinahe richtig segmentiert wird bis auf zwei Bereiche im oberen Teil. Die pfahlartigen Objekte im Bild werden verschwommen erkannt.

(Folie 33)

Besonders Interessant ist die Klassifizierung der Pfähle in der Punktwolke, die weitgehend richtig ist. Die verschwommenen Ränder auf dem segmentierten Bild bewirken, dass Teile der Straße falsch erkannt werden. Da Objekte im Vordergrund von größerem Interesse als der Hintergrund sind, ist dieses Verhalten durchaus positiv.

(Folie 34)

Aus den Experimenten lässt sich schlussfolgern, dass es einerseits ein Trade-Off zwischen Qualität der Ergebnisse und der Rechenzeit des Prozesses gibt. Die Ausgabe des Netzes kann andererseits auch durch Anpassung des Trainings, z.B. Nachtraining mit eigenen Daten verbessert werden. Außerdem ist eine hohe Varietät in den Trainingsdaten wünschenswert. Idealerweise bevorzugt das Netz bei der Bildsegmentierung Vordergrundobjekte. Als größtes Problem des verwendeten Neuronalen Netzes hat sich Überanpassung herausgestellt.

(Folie 35)

Danke für Ihre Aufmerksamkeit.